Theoretical Computer Science 412 (2011) 1646–1651

Contents lists available at ScienceDirect

Theoretical Computer Science

journal homepage: www.elsevier.com/locate/tcs

A well-mixed function with circuit complexity $5n$: Tightness of the Lachish–Raz-type bounds☆

Kazuyuki Amano [a,*], Jun Tarui [b]

[a] Department of Computer Science, Gunma University, Tenjin 1-5-1, Kiryu, Gunma 376-8515, Japan

[b] Department of Information and Communication Engineering, University of Electro-Communications, Chofu, Tokyo 182-8585, Japan

ARTICLE INFO

Keywords:
Boolean functions
Circuit complexity
Upper bound

ABSTRACT

A Boolean function on n variables is *k-mixed* if any two distinct restrictions fixing the same set of k variables induce distinct functions on the remaining $n - k$ variables. We give an explicit construction of an $(n - o(n))$-mixed Boolean function whose circuit complexity over the basis U_2 is $5n + o(n)$. This shows that a lower bound method for the size of a U_2-circuit that applies to arbitrary well-mixed functions, which yields the largest known lower bound of $5n - o(n)$ for the U_2-circuit size (Iwama, Lachish, Morizumi and Raz [STOC01, MFCS02]), has reached the limit.

1. Introduction and overview

A Boolean function on n variables is *k-mixed* if any two distinct restrictions fixing the same set of k variables induce distinct functions on the remaining $n - k$ variables. The notion of a k-mixed Boolean function, introduced by Jukna [9] (see also [16, p. 137]), plays an important role in deriving lower bounds in several computational models. The largest known $5n - o(n)$ lower bound [7,8] for the size of a Boolean circuit over the basis U_2 applies to *any* $(n - o(n))$-mixed Boolean function. The basis U_2 is the set of all Boolean functions over two variables except for the XOR function and its complement. It is also known that the size of any read-once branching program computing a k-mixed Boolean function is at least 2^k (see e.g., [14,12]). This result was used to show the existence of a function in P having a read-once branching program size not smaller than $2^{n-O(\sqrt{n})}$ [12] (see also [2] for an improved result).

In this paper, we focus on the complexity of Boolean circuits over the basis U_2. Deriving a good lower bound for an explicit Boolean function in such a general circuit model is one of the central problems in computer science. In 1991, Zwick [17] gave a lower bound of $4n-O(1)$ for a certain family of symmetric Boolean functions. After a decade, Lachish and Raz [11] introduced and considered a new family of Boolean functions, which they called (n, k)-strongly-two-dependent functions, and gave an improved lower bound of $4.5n - o(n)$. Shortly afterward, Iwama and Morizumi [7] proved a lower bound of $5n - o(n)$ for the same family of Boolean functions (see also [8]); this bound of $5n - o(n)$ is the largest known lower bound for the U_2-circuit size. It is easily seen that any k-mixed function is $(n, k - 2)$-strongly-two-dependent, and thus the lower bound of $5n - o(n)$ above applies to any k-mixed function with $k = n - o(n)$.

Iwama and Morizumi [7] improved the lower bound from $4.5n$ to $5n$ by refining the analysis of Lachish and Raz [11] and in particular using fairly long case analyses. Can we further improve the lower bound by yet finer analysis? The main result of this paper is that, somewhat surprisingly, the answer is negative: We show that there exists a well-mixed function with

☆ A preliminary version of this paper has appeared in Proceedings of the 5th Annual Conference on Theory and Applications of Models of Computation (TAMC 08), LNCS 4978, pp. 342–350 (2008).

* Corresponding author.
E-mail addresses: amano@cs.gunma-u.ac.jp (K. Amano), tarui@ice.uec.ac.jp (J. Tarui).

0304-3975/$ – see front matter © 2010 Elsevier B.V. All rights reserved.
doi:10.1016/j.tcs.2010.12.040

circuit complexity $5n + o(n)$. More precisely, we give an explicit construction of a Boolean function f_n on n variables with the following two properties:

(i) f_n is k-mixed for $k = n - t(n)$, where $t(n) = \omega(\sqrt{n}\log^2 n)$.
(ii) f_n can be computed by a circuit of size $5n + o(n)$ over the basis U_2.

From (i), it follows that any U_2-circuit for f_n has size at least $5n - o(n)$, and thus the U_2-circuit complexity of f_n is $5n \pm o(n)$. The result also shows that the lower bound method that applies to arbitrary k-mixed functions developed by Iwama, Lachish, Morizumi and Raz [8,11,7] has reached the limit.

A simple and explicit construction of an $(n - 3\sqrt{n})$-mixed Boolean function was given by Savický and Žák [12]. Their function is of the form $h(x) = x_{\phi(x)}$, where $\phi : \{0, 1\}^n \to \{1, 2, \ldots, n\}$ is a kind of weighted sum of inputs. The function we consider is a modification of their function. We modify their function so that the function can be shown to be computable by U_2-circuits of size $5n$, and at the same time the function remains highly mixed.

The paper is organized as follows. In Section 2, we give some preliminaries. In Section 3, we give a definition and an analysis of our function. Finally, we give some concluding remarks in Section 4.

2. Preliminaries

The set of all natural numbers is denoted by $\mathbb{N}$. For $n \in \mathbb{N}$, $[n]$ denotes the set $\{1, 2, \ldots, n\}$. For a binary sequence $x = x_{k-1} \cdots x_0$, $(x)_2$ denotes the integer represented by x, i.e., $(x)_2 = \sum_{i=0}^{k-1} 2^i x_i$.

Throughout the paper, we consider Boolean functions on the variable set $X_n = \{x_1, \ldots, x_n\}$. We let B_2 denote the set of all (sixteen) Boolean functions over two variables, and let U_2 denote $B_2 - \{\oplus, \equiv\}$, i.e., U_2 contains all Boolean functions over two variables except for the XOR function and its complement. For a basis $B \subseteq B_2$, *a Boolean circuit* over the basis B is a directed acyclic graph with nodes of in-degree 0, 1 or 2. Nodes of in-degree 0 are called *input-nodes*, and each one of them is labeled by a variable in X_n or a constant 0 or 1. Nodes of in-degree 1 or 2 are called *gate-nodes*, and each one of them has one or two inputs and an output, and is labeled by a function in B. For a basis B and for a Boolean function f, the circuit complexity of f over the basis B, denoted by $\mathsf{Size}_B(f)$, is the minimum number of gate-nodes in a circuit over the basis B that computes f.

Let f be a Boolean function on X_n. A *partial assignment* is a map $\sigma : X_n \to \{0, 1, *\}$, where $\sigma(x_i) = 0$ or 1 means that the input variable x_i is fixed to constant 0 or 1, and $\sigma(x_i) = *$ means that x_i remains free. For a partial assignment σ, the *support* of σ is the set of variables mapped to 0 or 1 by σ. The function obtained from f by applying the partial assignment σ is denoted by $f|_\sigma$. Note that $f|_\sigma$ is a function of the variables x_i for which $\sigma(x_i) = *$.

All logarithms in the paper are to base 2.

3. Function: mixedness and tight upper bound

In this section, we give an explicit construction of a well mixed Boolean function whose circuit complexity over the basis U_2 is $5n \pm o(n)$.

Definition 1. Let $k \in \mathbb{N}$. A Boolean function f on X_n is *k-mixed* if for every $V \subseteq X_n$ such that $|V| = k$ and for any two distinct partial assignments α, β with support V, the functions obtained from f by applying α and β are distinct, i.e., $f|_\alpha \neq f|_\beta$.

We now define our function. Given an arbitrary natural number n, we want to construct a highly mixed function f_n on n variables. Our function is similar to the function introduced by Savický and Žák [12], and our function is the pointer function of the form $f_n(x_1, \ldots, x_n) = x_z$. The only modification from their construction is in the definition of an index z. We define z by a weighted sum of *block parities* of input variables.

Let p be a prime such that $n \leq p < 2n$. By the Bertrand–Chebyshev theorem (see e.g., [6, p.373]), such a prime exists. Define $w_n : \{0, 1, \ldots\} \to [n]$ so that $w_n(s)$ is the residue of s modulo p, if this residue lies in $[n]$, and is 1 otherwise.

Put $b = \lceil \frac{n}{\lceil \log^2 n \rceil} \rceil$. We split the interval $[n]$ into b blocks $D_1, \ldots, D_b$ so that each block D_j $(1 \leq j < b)$ contains consecutive integers and the last block D_b contains $n - (b-1)\lceil \log^2 n \rceil$ consecutive integers, i.e., $n - (b-1)\lceil \log^2 n \rceil + 1, \ldots, n$. Thus, an index $i \in [n]$ is in D_j, where $j = \lceil \frac{i}{\lceil \log^2 n \rceil} \rceil$. We call this j the *weight* of an index i and denote it by $\mathrm{wgt}(i)$. For $x \in \{0, 1\}^n$ and $i \in [b]$, let $\mathsf{PAR}(x, i)$ denote the parity of variables in the i-th block, i.e.,

$$\mathsf{PAR}(x, i) = \bigoplus_{j \in D_i} x_j.$$

Definition 2. For every $n \in \mathbb{N}$, the Boolean function $f_n : \{0, 1\}^n \to \{0, 1\}$ is defined as $f_n(x) = x_z$, where

$$z = w_n\left(\sum_{i=1}^{b} i \cdot \mathsf{PAR}(x, i)\right). \tag{1}$$

If one replaces Eq. (1) with $z := w_n(\sum_{i=1}^{n} i \cdot x_i)$, then one obtains the function of Savický and Žák [12].

We will show that the function f_n defined above is k-mixed for $k = n - t(n)$, where $t(n) = \omega(\sqrt{n}\log^2 n)$ (Theorem 1), and then show that the circuit complexity of f_n over the basis U_2 is $5n + o(n)$ (Theorem 3).

3.1. Mixedness of f_n

Theorem 1. *The function f_n is $(n-t)$-mixed for $t = t(n) = \omega(\sqrt{n}\log^2 n)$.*

Remark. In the definition of f_n, we can choose the size of each block to be $\Omega(\log^{1+\epsilon} n)$ for any $\epsilon > 0$, and obtain the same conclusion for $t(n) = \omega(\sqrt{n}\log^{1+\epsilon} n)$ in Theorem 1 and the same $5n$ upper bound in Theorem 3. We pick and fix $\epsilon = 1$ for simplicity. We can *not* draw the same conclusion if we decrease the block size down to $O(\log n)$ since this would increase the size of a circuit nonnegligibly in the construction of step (iii) in the proof of Theorem 3.

The outline of the proof of Theorem 1 is as follows. Consider any two different partial assignments u and v such that the sets of $*$-variables in u and in v are same and this same set has size $\omega(\sqrt{n}\log^2 n)$. This ensures that $\omega(\sqrt{n})$ blocks contain at least one $*$-variable since each block has $O(\log^2 n)$ variables. Our goal is to find an assignment x^* on these $*$-variables such that the two pointer functions $f|_u(x^*)$ and $f|_v(x^*)$ point to variables having different values. We will show that this is always achievable by using the following theorem due to da Silva and Hamidoune [13] (see [1] for a simplified proof), which was also used by Savický and Žák [12].

Theorem 2 (*da Silva and Hamidoune [13]*)**.** *Let p be a prime and let z and h be two integers. Let $h \le z \le p$, and let $A \subseteq Z_p$ be such that $|A| = z$. Let B be the set of all sums of h distinct elements of A. Then $|B| \ge \min(p, hz - h^2 + 1)$.*

In particular, we will use the following consequence of this theorem: If $h(|A| - h) \ge p$, then every element in Z_p can be written as a sum of h elements of A. (Instead of using the result of da Silva and Hamidoune [13], we can also derive our result using the earlier result of Erdös and Heilbronn [5], which says that if $|A| \ge \Omega(\sqrt{p})$, then every element in Z_p can be written as a sum of some distinct elements of A.)

Our argument will be basically similar to the proof of the result by Savický and Žák [12] mentioned above.

Proof of Theorem 1. Let $I \subseteq [n]$ with $|I| = n - t$, and let u and v be two distinct partial assignments that fix all the variables whose indices are in I. To show that $f_n|_u \ne f_n|_v$, it suffices to show that there are two total assignments x and y such that $f_n(x) \ne f_n(y)$, where x and y are extensions of u and v, respectively, and x and y coincide on $\bar{I} = [n]\backslash I$.

Let J be an arbitrary maximal subset of $\bar{I}$ such that every two elements of J have distinct weights, i.e., $\mathrm{wgt}(i) \ne \mathrm{wgt}(j)$ for every $i, j \in J$ with $i \ne j$. Note that $|J| = \omega(\sqrt{n})$ since $t = \omega(\sqrt{n}\log^2 n)$ and there are at most $O(\log^2 n)$ indices having the same weight.

Let u^0 and v^0 be the total assignments obtained from u and v by assigning the value 0 to every position in $\bar{I}$.

Partition the set of indices J into the two disjoint sets J_s and J_d defined as follows:

$$J_s = \{j \in J \mid \mathrm{PAR}(u^0, \mathrm{wgt}(j)) = \mathrm{PAR}(v^0, \mathrm{wgt}(j))\};$$
$$J_d = \{j \in J \mid \mathrm{PAR}(u^0, \mathrm{wgt}(j)) \ne \mathrm{PAR}(v^0, \mathrm{wgt}(j))\}.$$

We divide the proof of the theorem into two cases depending on the sizes of J_s and J_d. In the rest of the proof, the symbol $\equiv$ denotes the congruence modulo p.

Case A $|J_s| \ge |J_d|$.

In this case we have $|J_s| = \omega(\sqrt{n})$. Put $S(u) = \sum_{i=1}^{b} i \cdot \mathrm{PAR}(u^0, i)$ and put $S(v) = \sum_{i=1}^{b} i \cdot \mathrm{PAR}(v^0, i)$. Let Δ be the residue of $S(u) - S(v)$ modulo p.

Case A-1 $\Delta \not\equiv 0$.

Extend u and v to u' and v' by setting all positions in $\bar{I}\backslash J_s$ to the value 0, and further fixing one or two positions in J_s as follows: Choose any $j \in J_s\backslash\{1\}$, and put $\ell = w_n(j + \Delta)$. Since $\ell \equiv j + \Delta \not\equiv j$ or $\ell = 1 \ne j$, we have $j \ne \ell$. If $\ell \in J_s$, in u' and v' set the position j to 0 and set the position ℓ to 1. This ensures that $u'_j = v'_j \ne u'_\ell = v'_\ell$. If $\ell \notin J_s$, set the position j of both u' and v' in such a way that $u'_j = v'_j \ne v'_\ell$.

We have at least $|J_s| - 2 = \omega(\sqrt{n})$ positions that are still unspecified in u' and in v'. Let A by the set of such unspecified positions. For an index $j \in [n]$, define the *contribution* of j, denoted by c_j, as follows:

$$c_j = \begin{cases} \mathrm{wgt}(j), & \text{if } \mathrm{PAR}(u^0, \mathrm{wgt}(j)) = 0; \\ p - \mathrm{wgt}(j), & \text{if } \mathrm{PAR}(u^0, \mathrm{wgt}(j)) = 1. \end{cases}$$

Note that setting the value 1 to an unassigned variable x_j increases the total weight of x (i.e., $\sum_{i=1}^{b} i \cdot \mathrm{PAR}(x, i)$) by c_j. Since $\mathrm{wgt}(j) \le b = \lceil \frac{n}{\lceil \log^2 n \rceil} \rceil < p/2$ for every j, all indices in A have distinct contributions. Let $h = \lfloor |A|/2 \rfloor$. Since $|A|h - h^2 + 1 = \omega(n) \ge p$, by Theorem 2 there is a set $H \subseteq A$ of size h such that

$$\sum_{i \in H} c_i \equiv j - \sum_{i=1}^{b} i \cdot \mathrm{PAR}(u', i).$$

Let x be the extension of u' such that $x_i = 1$ for every $i \in H$ and $x_i = 0$ for every $i \in A \backslash H$. Then, we have

$$\sum_{i=1}^{b} i \cdot \mathsf{PAR}(x, i) \equiv j.$$

It follows that $f_n(x) = u'_j$. Let y be the assignment extending v so that x and y coincide on J_s, i.e., $x_i = y_i$ for $i \in J_s$. It follows that

$$\omega_n\left(\sum_{i=1}^{b} i \cdot \mathsf{PAR}(y, i)\right) = \omega_n\left(\sum_{i=1}^{b} i \cdot \mathsf{PAR}(x, i) + \Delta\right) = \ell,$$

and hence $f_n(y) = v'_\ell \neq f_n(x)$.

Case A-2 $\Delta \equiv 0$.

Fix any $j \in I$ satisfying $u_j \neq v_j$. Since $|J_s| = \omega(\sqrt{n})$, by Theorem 2 there is an extension x of u such that $\sum_{i=1}^{b} i \cdot \mathsf{PAR}(x, i) \equiv j$. Let y be the assignment extending v so that x and y coincide on J_s. Then, we have $f_n(x) = u_j \neq v_j = f_n(y)$.

Case B $|J_d| > |J_s|$.

Partition the set J_d into the two disjoint sets $J_{d,0}$ and $J_{d,1}$ defined as follows:

$$J_{d,0} = \{j \in J_d \mid \mathsf{PAR}(u^0, \mathsf{wgt}(j)) = 0 \wedge \mathsf{PAR}(v^0, \mathsf{wgt}(j)) = 1\};$$
$$J_{d,1} = \{j \in J_d \mid \mathsf{PAR}(u^0, \mathsf{wgt}(j)) = 1 \wedge \mathsf{PAR}(v^0, \mathsf{wgt}(j)) = 0\}.$$

Without loss of generality, we can assume that $|J_{d,0}| \geq |J_{d,1}|$ (otherwise swap u with v). Note that $|J_{d,0}| \geq |J_d|/2 = \omega(\sqrt{n})$. Put $S(u) = \sum_{i=1}^{b} i \cdot \mathsf{PAR}(u^0, i)$ and put $S(v) = \sum_{i=1}^{b} i \cdot \mathsf{PAR}(v^0, i)$. Let Δ be the residue of $S(v) - S(u)$ modulo p.

As in Case A-1, we extend u and v to u' and v' by setting all positions in $\bar{I} \backslash J_{d,0}$ to 0 and some positions in $J_{d,0}$ to a suitable value. What is different from Case A-1 is the fact that if we set a position in $J_{d,0}$ of weight k to the value 1, then the total weight of u increases by k and that of v *decreases* by k.

Let k be an arbitrary element in Z_p that satisfies $\omega_n(S(u)+k) \in J_{d,0} \backslash \{1\}$. Put $j = \omega_n(S(u)+k)$, and put $\ell = \omega_n(S(v)-k) = \omega_n(S(u) + \Delta - k)$. Without loss of generality, we can assume that $j \neq \ell$. The reason we can assume so is as follows: Since $j = \ell$ only when $S(u) + k \equiv S(u) + \Delta - k$, at most one choice of k yields $j = \ell$; we can avoid such a choice of k since $|J_{d,0}|$ is sufficiently large. If $\ell \in J_{d,0}$, in u' and v' set the position j to 0 and set the position ℓ to 1. If $\ell \notin J_{d,0}$, set the position j of both u' and v' in such a way that $u'_j = v'_j \neq v_\ell$. In both cases further fix all positions in $\bar{I} \backslash J_{d,0}$ of x and y by the value 0.

The rest of the proof is analogous to Case A-1. Since there are at least $|J_{d,0}| - 2 = \omega(\sqrt{n})$ unassigned positions, by Theorem 2 we can always extend u' to x so that $\sum_{i=1}^{b} i \cdot \mathsf{PAR}(x, i) \equiv S(u) + k$. Let y be the total assignment extending v' in the same way as x extends u'. Then, $\sum_{i=1}^{b} i \cdot \mathsf{PAR}(y, i) \equiv S(v) - k$, and therefore $f_n(x) = x_j \neq y_\ell = f_n(y)$. This completes the proof of Theorem 1. □

3.2. $5n$ upper bound for f_n

Now we proceed to the design and analysis of a circuit computing our function. We use a circuit called a *decoder* defined as follows.

Definition 3. An n-to-2^n decoder is a circuit computing the function $\mathsf{Decode}_n : \{0,1\}^n \to \{0,1\}^{2^n}$; it takes an n-bit binary input x and outputs $d_0 d_1 \cdots d_{2^n-1}$ such that $d_i = 1$ iff $(x)_2 = i$.

It is well-known that $\mathsf{Size}_{U_2}(\mathsf{Decode}_n) = 2^n + O(n2^{n/2})$ (see e.g., [15, p. 75]).

Theorem 3. *The function f_n can be computed by a circuit of size at most $5n + o(n)$ over the basis U_2.*

Proof. We break down the computation of f_n into five steps as follows :

(i) Compute $\mathsf{PAR}(x, i)$ for each $i = 1, \ldots, b$. Recall that b denotes the number of blocks for the input variables and that $b = O(n/\log^2 n)$.
(ii) Compute the binary representation of $i \cdot \mathsf{PAR}(x, i)$ for each $i = 1, \ldots, b$.
(iii) Compute the binary representation of $\sum_{i=1}^{b} i \cdot \mathsf{PAR}(x, i)$.
(iv) Compute the binary representation of $z = w_n\left(\sum_{i=1}^{b} i \cdot \mathsf{PAR}(x, i)\right)$.
(v) Output x_z.

Recall that the parities $\mathsf{PAR}(x, i)$ for $i = 1, \ldots, b$ are computed on disjoint blocks D_i of variables. In step (i), since a parity gate can be implemented by using three U_2-gates ($g_1 = \overline{x_1} \wedge x_2, g_2 = x_1 \wedge \overline{x_2}, g_3 = x_1 \oplus x_2 = g_1 \vee g_2$), all $\mathsf{PAR}(x, i)$'s can be computed by a circuit of size $3(n - b) < 3n$ over the basis U_2.

Let $(i_q, \ldots, i_1)$ be the binary representation of an $i \in [b]$, where $q = O(\log n)$. Then the binary representation of i·PAR(x, i) is obviously

$$(i_q \wedge \mathsf{PAR}(x, i), \ldots, i_1 \wedge \mathsf{PAR}(x, i)). \tag{2}$$

Since i_j's do not depend on an input, i_j's can be considered as constants, and we need no gates in step (ii); in fact, we can replace $i_j \wedge \mathsf{PAR}(x, i)$ in Eq. (2) by the constant 0 for each j with $i_j = 0$, and by $\mathsf{PAR}(x, i)$ for each j with $i_j = 1$.

In step (iii), we need $b - 1$ $O(\log n)$-bit adders, which can be realized using at most $b \cdot O(\log n) = o(n)$ gates over the basis U_2, since the addition of two k bit numbers can be implemented by a circuit of size $O(k)$ (see, e.g., [15]).

In step (iv), we only need several basic arithmetic operations on $O(\log n)$-bit numbers. The number of gates needed is obviously a polynomial in $O(\log n) = o(n)$.

In step (v), we use the *n-way multiplexer* (a.k.a. *storage access function*) M_n whose definition is as follows: Let $q = \lceil \log n \rceil$. The function M_n takes $q + n$ binary inputs and is defined as

$$M_n(z_1, \ldots, z_q, x_0, \ldots, x_{n-1}) = x_{(z_q \cdots z_1)_2}.$$

If $(z_q \cdots z_1)_2 \geq n$, then the output of M_n is unspecified. It is well known that M_n can be computed by a circuit of size $2n + o(n)$ over the basis U_2 when n is a power of two [10] (see also [15, p. 77]). We describe the construction due to [10], and verify that $\mathsf{Size}_{U_2}(M_n) = 2n + o(n)$ for every n.

By using an identity in [10] (or [15, p. 77]), we have

$$M_n(z_1, \ldots, z_q, x_0, \ldots, x_{n-1}) = M_{2^{\lfloor q/2 \rfloor}}(z_1, \ldots, z_{\lfloor q/2 \rfloor}, x'_0, \ldots, x'_{2^{\lfloor q/2 \rfloor}-1}),$$

where, for each $i = 0, \ldots, 2^{\lfloor q/2 \rfloor} - 1$,

$$x'_i = \bigvee_{t=0}^{2^{\lceil q/2 \rceil}-1} (d_t \wedge x_{2^{\lceil q/2 \rceil} i + t}), \tag{3}$$

and d_t is the t-th output of $\mathsf{Decode}_{\lceil q/2 \rceil}$ applied to $(z_{\lfloor q/2 \rfloor + 1}, \ldots, z_q)$. Here we put $x_j = 0$ for every $j \geq n$ in Eq. (3). Thus, M_n can be realized using n AND gates and at most n OR gates in addition to $O(2^{q/2}) = o(n)$ gates used to compute the function $M_{2^{\lfloor q/2 \rfloor}}$ and $\mathsf{Decode}_{\lceil q/2 \rceil}$.

Overall, the total size of our circuit for the function f_n is $(3 + 2)n + o(n) = 5n + o(n)$. This completes the proof of Theorem 3. □

In summary, we have the following:

Theorem 4. *There is a sequence of Boolean functions $\{f_n\}_n$, which is given by Definition 2, such that* (i) *f_n is a Boolean function on n variables,* (ii) *f_n is k-mixed for $k = n - t(n)$, where $t(n) = \omega(\sqrt{n} \log^2 n)$, and* (iii) *the circuit complexity of f_n over the basis U_2 satisfies the following:*

$$5n - o(n) \leq \mathsf{Size}_{U_2}(f_n) \leq 5n + o(n).$$

4. Concluding remarks

In the paper, we gave an explicit construction of an $(n - o(n))$-mixed Boolean function with circuit complexity $5n \pm o(n)$. Our results shows that a lower bound method for the size of a U_2-circuit that applies to arbitrary well mixed functions has reached the limit. As pointed out by one of reviewers of this paper, over the complete basis B_2, our function can be computed by a circuit of size $3n + o(n)$ since we only need n gates instead of $3n$ gates in step (i) in the construction. This again asymptotically matches the largest known lower bound of $3n - o(n)$ for the B_2-circuit size of an explicit Boolean function due to Blum [3,15]; and one cannot obtain an improvement beyond $3n$ if one only considers well-mixed functions.

Our result shows a limitation of the gate elimination method under one particular setting. One can pursue more results saying that the gate elimination method cannot establish X. It would be particularly interesting if one can show that (some form of) gate elimination methods are inherently too weak to derive a nonlinear lower bound under some setting.

Acknowledgements

We would like to thank the anonymous reviewers for their comments that greatly improved the presentation of this paper.

References

[1] N. Alon, M.B. Nathanson, I. Ruzsa, The polynomial method and restricted sums of congruence classes, J. Number Theory 56 (1996) 404–417.
[2] A. Andreev, J. Baskakov, A. Clementi, J. Rolim, Small pseudo-random sets yields hard functions: new tight explicit lower bounds for branching programs, in: Proc. 26th ICALP (1999) 179–189 (Also in ECCC TR97-053 (1997)).

[3] N. Blum, A Boolean function requiring $3n$ network size, Theoret. Comput. Sci. 28 (1984) 337–345.
[4] C.C. Forster, F.D. Stockton, Counting responders in an associative memory, IEEE Trans. Comput. C-20 (12) (1971) 1580–1583.
[5] P. Erdös, H. Heilbronn, On the addition of residue classes mod p, Acta Arithmetica IX (1964) 149–159.
[6] G.H. Hardy, E.M. Wright, An Introduction to the Theory of Numbers, fifth edition, Oxford University Press, 1978.
[7] K. Iwama, H. Morizumi, An explicit lower bound of $5n - o(n)$ for boolean circuits, in: Proc. 27th MFCS, 2002, pp. 353–364.
[8] K. Iwama, O. Lachish, H. Morizumi, R. Raz, An Explicit Lower Bound of $5n - o(n)$ for Boolean Circuits", Manuscript, 2005. Available at http://www.wisdom.weizmann.ac.il/~ranraz/publications/, preliminary versions are [11] and [7]).
[9] S. Jukna, Entropy of contact circuits and lower bounds on their complexity, Theoret. Comput. Sci. 57 (1988) 113–129.
[10] P. Klein, M. Paterson, Asymptotically optimal circuit for a storage access function, IEEE Trans. Comput. 29 (8) (1980) 737–738.
[11] O. Lachish, R. Raz, Explicit lower bound of $4.5n - o(n)$ for boolean circuits, in: Proc. 33rd STOC, 2001, pp. 399-408.
[12] P. Savický, S. Žák, A Large lower bound for 1-branching programs, in: ECCC TR96-036 Rev.01, 1996.
[13] J. Dias da Silva, Y. Hamidoune, Cyclic spaces for grassmann derivatives and additive theory, Bull. London. Math. Soc. 26 (1994) 140–146.
[14] J. Simon, M. Szegedy, A new lower bound theorem for read only once branching programs and its applications, in: Advances in Computational Complexity Theory, in: DIMACS Series, vol. 13, 1993, pp. 183–193.
[15] I. Wegener, The Complexity of Boolean Functions, Willey-Teubner, 1987.
[16] I. Wegener, Branching Programs and Binary Decision Diagrams, SIAM, 2000.
[17] U. Zwick, A $4n$ lower bound on the combinatorial complexity of certain symmetric boolean functions over the basis of unate dyadic boolean functions, SIAM J. Comput. 20 (1991) 499–505.